REMARKS

Claims 1-6, 8-13, 15, and 17-18 are pending. Applicant respectfully traverses and requests reconsideration.

I. Summary of the Examiner's Objections/Rejections

Claims 1-6, 8-13, 15, and 17-18 are rejected under 35 U.S.C. §102(b) based on Favor "Favor" (WO 97/13194).

FAVOR

Common x86 instructions are converted by instruction **decode hardware** to operations in an internal RISC86 instruction set. (Favor, page 3 lines 36-37, emphasis added). One problem, described by Favor, with the usage of lookup ROM for decoding x86 instructions is that the process of accessing a microprogram control store is inherently slower and less efficient than hardwired translation of instructions. (Favor, p. 1, lines 28-30). Favor describes an internal RISC type instruction format that facilitates translation of very large number of CISC-type instructions into a small number of RISC-type operations. (Favor, p. 1, lines 38-39). An internal instruction format permits more common CISC-type instructions to be converted using hardware logic, as compared to conversion via lookup ROM. (Favor, p. 1, lines 41-42). The instruction set includes a plurality of instruction codes arranged in a fixed bit length structure. (Favor, p. 2, lines 11-12). A regular structure greatly reduces circuit complexity and size and substantially increases efficiency. (Favor, p. 2, lines 28-30).

Instructions from main memory 130 are loaded into instruction cache 214 via a predecoder 270 for anticipated execution. (Favor, p. 4, lines 9-11). The predecoder 270 generates predecode bits that are stored in combination with instruction bits in the instruction cache 214. (Favor, p. 4, lines 11-12).

X86 instructions typically include an opcode followed by a modr/m byte. (Favor, p. 34, lines 28). The modr/m byte designates an indexing type or register number to be used in the instruction. (Favor, p. 34, lines 28-29). Figure 6A is a register operation (RegOp) field 610 encoding graphic that illustrates various fields in the RegOp format. (Favor, p. 35, lines 1-2). The RegOp field 610 also includes a single-bit set status (SS) field 624 at bit location [9].... (Favor, p. 35, lines 7-8). For Ops in which the set status (SS) field 624 is set to 1, indicating that this Op does modify flags, the extension field (EXT) 614 specifies four status modification bits designating the groups of flags that are modified by the Op. (Favor, p. 36, lines 25-27).

Decoupling of condition code handling from operation type, using the independent TYPE 612 and set status (SS) field 624 allows some operations to be defined which do not update the flags. (Favor, p. 38, lines 7-9). The EXT field 614 is used to update condition flags including six flags corresponding to x86 flags and two emulation flags. (Favor, p. 38, lines 1-2)

35 U.S.C. §102(b) Rejections

Claims 1-6, 8-13, 15 and 17-18 are rejected under 35 U.S.C. §102(b) over Favor "Favor" (WO 97/13194). A claim is anticipated only if each and every element arranged as required by the claim is found in a single prior art reference. MPEP 2131.

Independent Claims 1 and 10

As to claims 1 and 10, these claims require, among other things, "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code."

The Favor language as cited at p. 36, lines 25-27 which states "[f]or Ops in which the set status (SS) field 624 is set to 1, indicating that this Op does modify flags, the extension field (EXT) 614 specifies four status modification bits designating the groups of flags that are modified by the Op" is limited to merely indicating the that this Op does modify flags for Ops in

which the set status (SS) field 624 is set to 1, rather than "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Applicants would like to point out the distinction between merely indicating "that this Op does modify flags for Ops in which the set status (SS) field 624 is set to 1" and "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Nevertheless, the office action states "Inherently, Favor's system must determine whether bit 624 is set when processing 612-type instructions and then will update at least one flag when the bit is set." However, the Office Action fails to show how or what portion of Favor's system performs "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Applicants would like to point out the distinction between the conditions "determine whether bit 624 is set when processing 612-type instructions" and "in response to executing the operational code." Further, Favor is directed to solving an entirely different problem, namely a regular structure that greatly reduced circuit complexity and size and substantially increases efficiency (Favor, page 2, lines 28-30). As a result, Favor has no need for "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Accordingly, Applicants request a showing of where Favor teaches "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code."

Rather than teach "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code" Favor teaches "Common x86 instructions are converted by instruction **decode hardware** to operations in an internal RISC86 instruction set." (Favor, page 3 lines 36-37, emphasis added). As a result,

Favor teaches that the conversion is performed in decode hardware rather than by hardware that would execute the instruction and therefore Favor teaches away from "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." For example, Favor teaches various condition codes (i.e. "WRFLG conditional Op", page 36 line 28; "condition codes" page 36 line 37 – page 37 line 18 "branch conditions" page 37 lines 26-35). Favor explicitly teaches that the *decoder* then changes the branch prediction as described at page 37 line 35-36 rather than "in response to executing the operational code." Favor also teaches generating predecode bits in predecoder 270. (Favor, p. 4, lines 9-11). Consequently, rather than describe "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code", Favor instead teaches indicating the that this Op does modify flags as a result of decode as opposed to "in response to executing the operational code." As a result, Favor teaches away from the claims. [1]

The Favor language as cited at p. 35, lines 7-8 which states "[t]he RegOp field 610 also includes a single-bit set status (SS) field 624 at bit location [9]" is limited to a single-bit set status (SS) field 624 located within the RegOp field 610 rather than "receiving at least one instruction containing data representing operational code and data representing at least one flag modification enable bit." Applicants would like to point out, among other things, the distinction between "an instruction containing data representing operational code and data representing at least one flag modification enable bit" and "[t]he RegOp field 610 also includes a single-bit set status (SS) field 624." Since Favor requires that the RegOp field include the SS field, Favor

[1] A prior art reference must be considered in its entirety, i.e., as a whole, including portions that would lead away from the claimed invention. *W.L. Gore & Associates, Inc. v. Garlock, Inc.*, 721 F.2d 1540, 220 U.S.P.Q. 303 (Fed. Cir. 1983), *cert. denied*, 469 U.S. 851 (1984), M.P.E.P. 2141.02.

does not teach that the instruction contains data representing operational code and data representing at least one flag modification enable bit, as arranged in the claims.

The Favor language as cited at p. 38, lines 7-9 which states "decoupling of condition code handling from operation type, using the independent TYPE 612 and set status (SS) field 624 allows some operations to be defined which do not update the flags," which is limited to operations defined which do not update the flags "using the independent TYPE 612 and set status (SS) field 624" rather than "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Applicants would like to point out the distinction between "using the independent type 612 and set status field 624" and "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." The Applicants cannot find where the cited portion of Favor teaches each and every element as arranged in the claims, namely "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Accordingly, the Applicants request a showing of "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code." Consequently, as the Office Action has similarly ignored a principal limitation of Claims 1 and 10, namely "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code" and since the Office Action does not disclose how Favor teaches "determining whether the at least one flag modification enabled bit allows updating of at least one flag in response to executing the operational code" as defined in Claims 1 and 10, Applicants submit that Favor

does not anticipate the invention as defined in Claims 1 and 10. As a result, Favor fails to teach each and every element as arranged in the claims and therefore, the rejection is improper.

Dependent Claims 4 and 13

Applicants further submit that Claims 4 and 13 are also allowable in light of the presence of novel and non-obvious elements contained in Claims 4 and 13 that are not otherwise present in Claims 1 and 10 respectively. Applicant also submits that Claims 4 and 13 depend from Claims 1 and 10 respectively, as dependent therefrom, Claims 4 and 13 are allowable for at least the reasons Claims 1 and 10 respectively are allowable. Applicants request a showing of where Favor teaches "updating a flag register if the flag modification enable bit is set to allow modification of a flag in the flag register" as arranged in the claims.

Dependent Claim 5

Favor as cited teaches indicating the that this Op does modify flags for Ops in which the set status (SS) field 624 is set to 1, rather than "evaluating the flag modification enable bit to preserve flag bit settings for variable length instructions that are emulated using the fixed native instructions." Applicants would like to point out the distinction between merely indicating "that this Op does modify flags for Ops in which the set status (SS) field 624 is set to 1" and "evaluating the flag modification enable bit to preserve flag bit settings for variable length instructions that are emulated using the fixed native instructions." Applicants request a showing of where Favor teaches "evaluating the flag modification enable bit to preserve flag bit settings for variable length instructions that are emulated using the fixed native instructions."

Applicants further submit that Claim 5 is also allowable in light of the presence of novel and non-obvious elements contained in Claim 5 that are not otherwise present claim 1.

Applicant also submits that Claim 5 depends from Claim 1, as dependent therefrom, Claim 5 is allowable for at least the reasons claimed 1 is allowable.

Dependent Claims 9 and 18

As stated above, Favor, understood as cited, is limited to converting X86 instructions by decode hardware to operations in an internal RISC86 instruction set. Applicants cannot find where Favor as cited teaches among other things, "emulating unconverted variable length X86 instructions." Applicants submit that the Office Action fails to show where Favor teaches where length X86 instructions are unconverted. Accordingly, applicants request a showing of where Favor teaches:

> converting variable length X86 instructions to a plurality of native instructions wherein the plurality of native instructions include that at least one flag modification enable bit set to allow changing of non-native instruction flags in response to execution of the plurality of native instructions; and
>
> emulating unconverted variable length X86 instructions using a plurality of native instructions wherein the native instructions include the at least one flag modification enable bit set to prevent changing of non-native instruction flags in response to execution of the plurality of native instructions.

Applicants further submit that Claims 9 and 18 are also allowable in light of the presence of novel and non-obvious elements contained in Claims 9 and 18 that are not otherwise present in Claims 1 and 10 respectively. Applicant also submits that Claims 9 and 18 depend from Claims 1 and 10 respectively, as dependent therefrom, Claims 9 and 18 are allowable for at least the reasons Claims 1 and 10 respectively are allowable.

Dependent Claims 3, 6, 8, 11, 12, 15, and 17

Applicants further submit that Claims 3, 6, 7, 8, 11, 12, 15, and 17 are also allowable in light of the presence of novel and non-obvious elements contained in these Claims that are not otherwise present claims 1 and 10. Applicant also submits that these Claims depend from

Claims 1 and 10, as dependent therefrom, Claims 3, 6, 7, 8, 11, 12, 15, and 17 are allowable for at least the reasons claims 1 and are allowable.

Applicants respectfully request that the pending claims be allowed to issue. The Examiner is invited to contact the below-listed attorney if the Examiner believes that a telephone conference will advance the prosecution of this application.

Respectfully submitted,

Date: February 5, 2004

By: _____
Themi Anagnos, Reg. No. 47,388

Vedder, Price, Kaufman & Kammholz
222 North LaSalle Street
Chicago, Illinois 60601
PHONE: (312) 609-7970
FAX: (312) 609-5005